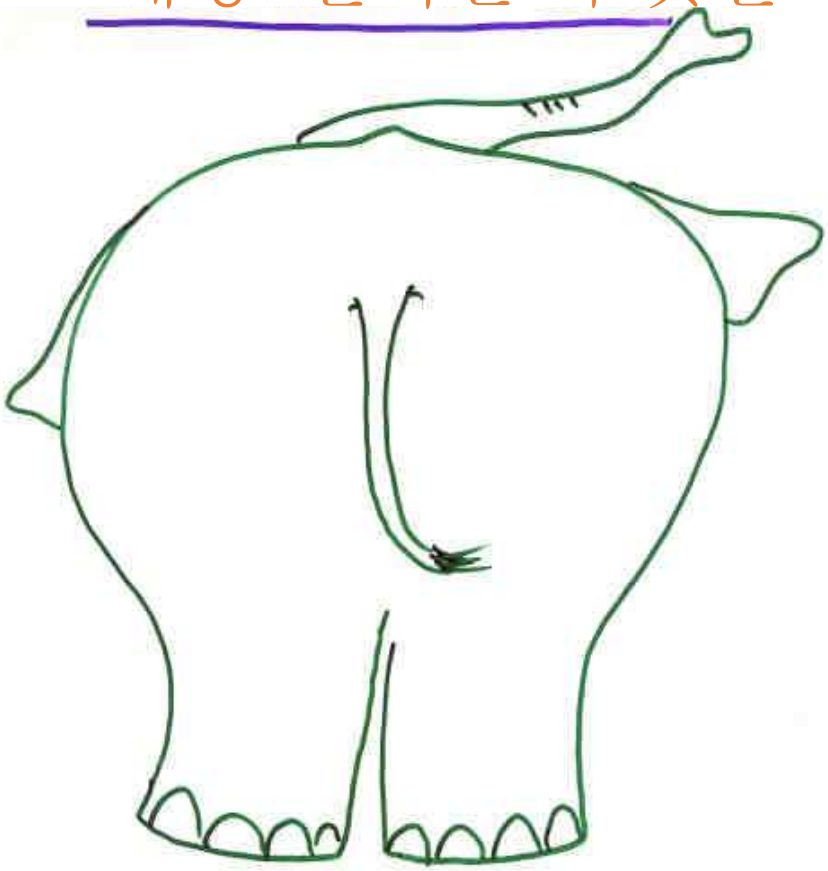
Programming Languages

Preliminaries

From prof. Kwangkeun Yi, SNU

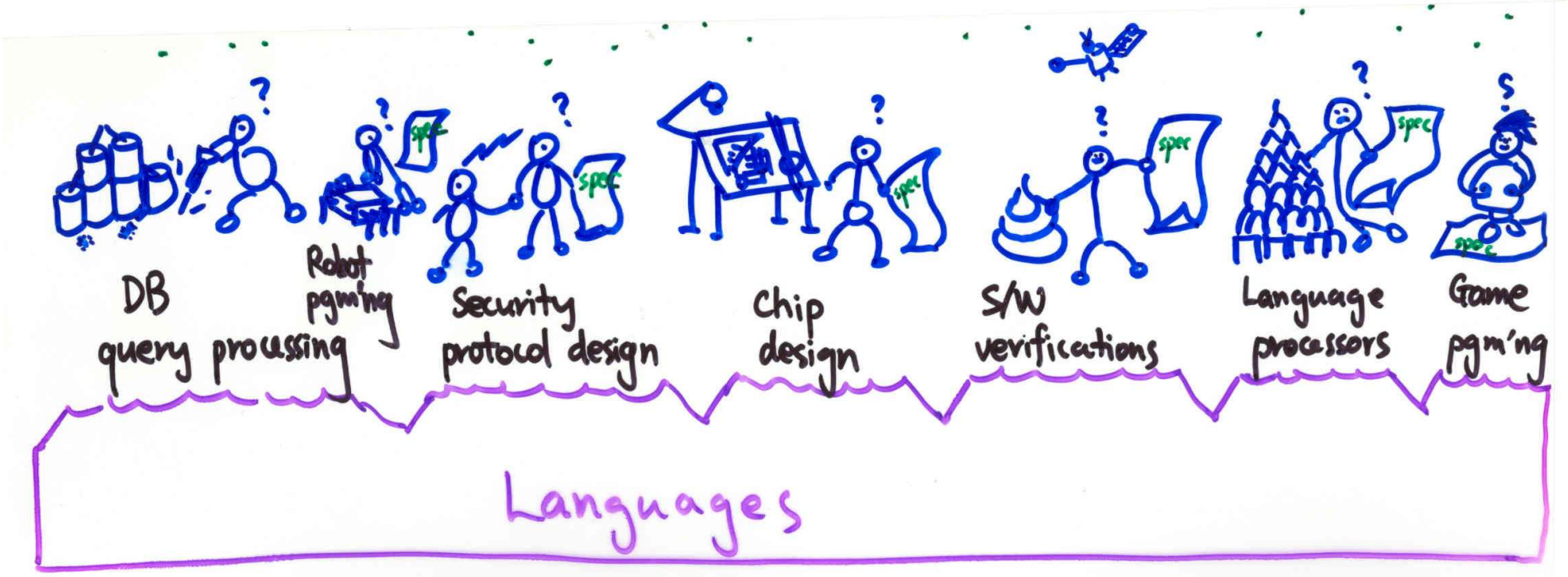
프로그래밍 언어란 무엇인가?

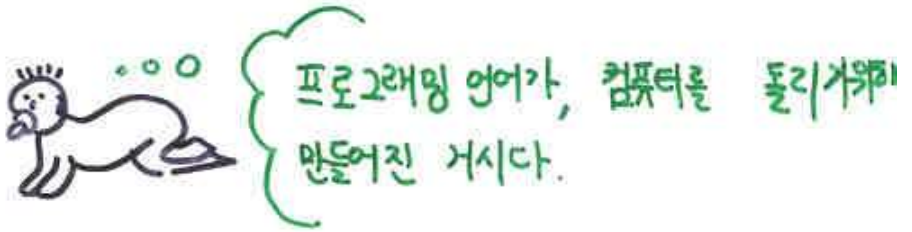


tools for instructing machines?
notation for algorithms?
means for expressing high-level designs?
tools for realization?


Logicians
Mathematicians


P.L.
P.L.


DB query processing
Robot pgm'ing
Security protocol design
Chip design
S/W verifications
Language processors
Game pgm'ng
spec
Languages


프로그래밍 언어가, 컴퓨터를 돌리기위해
만들어진 거시다.

프로그래밍 언어가, 컴퓨터를 돌리기위해
만들어진 거시다.

不绒气多

컴퓨터가, 프로그램을 돌리기위해
만들어진 것이다.

불성무물(不誠無物) - 『中庸』
우주라는 공간에 존재하는 모든 것은 자기 존재의 법칙을 가지고 있다. 그 법칙이 바로 성실할 '성(誠)' 자인데, 즉 세상의 모든 것은 결국 성실함을 통해 이루어졌고 성실함을 통해 존재하며 성실함을 통해 발전한다는 것이다.

『中庸』에 "성실함은 세상의 모든 것을 이루는 원리이며 성실함이 없다면 그 어떤 존재도 있을 수 없다." 라는 글이 있는데, 이런 뜻을 담은 사자성어가 바로
'불성무물(不誠無物)' 이다.

(프로그래밍) 언어 배우기

- "이렇게 쓰는 거야" : 문법 구조

겉모양 → Syntax

- "이게 이런 뜻이란다" : 의미 구조

속내용 → Semantics

문법과 의미 (겉과 속)이 과연 완전히 구분될 수는 없겠지만 ...

* abstract syntax
* semantic formalisms

To learn :

syntax . 문법 구조

abstract syntax . 문법 구조의 핵심만을 표현하는 방법

semantics . 의미구조

semantic formalisms . 의미구조를 정확히 표현하는 방법

warm-ups . 한번 해 보자 !

프로그래밍 언어를 공부하는 데 필요한 기본기의 첫 스텝

그래, 질문요!

근데, 질문요!

- 한 언어로 작성할 수 있는 프로그램은 대게 무한히 많지 않습니까?

근데, 질문요!

- 한 언어로 작성할 수 있는 프로그램은 대게 무한히 많지 않습니까?

- 무한히 많은 프로그램의 문법과 의미를 유한한 한 학기 동안 강의 할 수 있으시겠습니까?

무한한 물건을
유한한 방법으로
정의해 본 기억?

근데, 질문요 !

- 한 언어로 작성할 수 있는 프로그램은
  대개 무한히 많지 않습니까?

- 무한히 많은 프로그램의 문법과 의미를
  유한한 한 학기 동안 강의 할 수 있으시겠습니까?

* 모든 자연수 n 에 대해서

$$\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$$

이러한 등식을 증명해 본 적이 있습니까?

Question #1

프로그램의 개수는 얼마나 되나? 셀 수 있는가?

! 프로그램의 개수

= 튜링 머신의 개수

= ?

Key point #1

우리는
무한을 다룰 수 있는
유한한 도구가 필요하다.

! 무한한 수, 예) 자연수

-> 투박하게는 집합

-> 보다 엄밀하게는 귀납법

! 프로그래밍언어의 경우

-> "문법"

보다 상세한 정의와 설명은
형식언어 수업으로 연계...

Inductive Definition

귀납적인 정의

"나는 귀납한다
고로 전산학을 한다."

방법 I

· / 은 자연수이다.

· ✱ 가 자연수이면 /✱ 도 자연수이다.

방법 I′

$$\frac{}{/\text{은 자연수이다}} \qquad \frac{✱\text{는 자연수이다}}{/✱\text{는 자연수이다}}$$

방법 I (I′) 에 의해서 자연수라고 판명되는 것들만 모두 모은것이 자연수의 집합.

방법 II

$$N \to /, \quad N \to /N \qquad \text{or} \qquad N \to / \mid /N$$

방법 II 에 의해서 생성되는 것들만 모두 모은 것이 자연수의 집합.

귀납법

귀납적인 정의 : 모든 걸 유한한 방법으로 그리고 정확하게

문법

N → /
  | / N

의미

⟦/⟧ = 1

⟦/N⟧ = 1 + ⟦N⟧

/

//

///

////

귀납적인 정의 : 모든 걸 유한한 방법으로 그리고 정확하게

문법

$\mathbb{N} \to /$

$\quad | \quad / \mathbb{N}$

의미

$[\![/]\!] = 1$

$[\![/\mathbb{N}]\!] = 1 + [\![\mathbb{N}]\!]$

$\triangle \to \circ$

$\quad |$ 

$\quad |$

$\quad |$ 

$[\![\circ]\!] =$

$[\![$  $]\!] =$

$[\![$  $]\!] =$

$[\![$  $]\!] =$

귀납적인 정의 : 모든 걸 유한한 방법으로 그리고 정확하게

문법

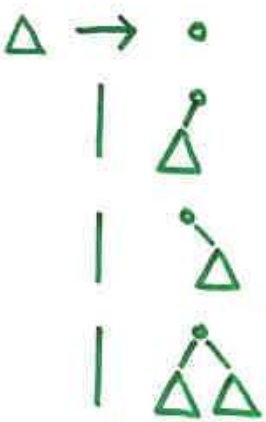
$\mathbb{N} \to /$

$\mid \; / \mathbb{N}$

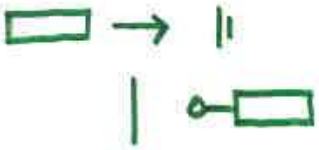
의미

$[\![/]\!] = 1$

$[\![/\mathbb{N}]\!] = 1 + [\![\mathbb{N}]\!]$

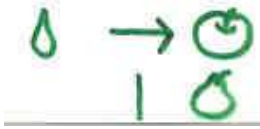


$[\![\circ]\!] =$



$[\![\;]\!] =$

$[\![\;]\!] =$



$[\![\;]\!] =$

$[\![\;]\!] =$

Question #2

Semantics가 정의되지 않은 프로그램의 경우 우리는 어떻게 해석해야 하는가?

! C 프로그래밍 언어의 경우 semantics는 완전한가?

! 다음 연산식은 문법적 오류인가?

x <= y <= z

! 또는 어떻게 해석해야 하는가?

! 누가 해석하는 것인가?

Question #3

```
 x <= y <= z
...


if (x > 10 && y-- > 5)
{
    ...
}
```

불분명한 semantics는
문제를 일으킬 수 있는
원인이 되는가?

Key point #2

표기하는 방법에 따라
나열된 기호와
그 의미는
일치하지 않을 수 있음

! Syntax와 Semantics는 쌍으로 구성되어 있음.

! Syntax와 Semantics를 정확하게 알아야 함.

! 몇몇 프로그래밍언어의 일부 요소는 semantics가 정의되지 않은 경우도 존재함.

프로그래밍 언어의 경우도 마찬가지입니다

e.g.) 정수식

0 -1 2

1+2 3*5 6**7 -3-6 7/2

문법 **의미**

E → Z
| E + E
| E - E
| E * E
| E / E
| E ** E